REMARKS

Applicant thanks Examiner for the detailed review of the application.

Claim Status

Claims 1, 3-4, 6, 11, and 21 have been amended.

Claims 5, 8, and 32 remain cancelled.

Claim Rejections -35 USC § 103(a)

The Office Action states:

> 5. Claims 1-4, 6-7, 9-31 and 33-41 are rejected under 35 U.S.C. 103(a) as being unpatentable over Chamdani et al. (US 2004/0073906 A1) in view of Hoogerbrugge et al. (US 6,615,333 B1) and further in view of Kranich (US 6,574,725 B1).

"The examiner bears the initial burden of factually supporting any prima facie conclusion of obviousness." MPEP § 2142. It is well established that *prima facie* obviousness is only established when three basic criteria are met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. *In re Vaeck*, 947 F.2d 488 (Fed. Cir. 1991) (MPEP 2144).

Applicant's claim 1 includes, "blocker logic to **prevent data associated with a speculative store instruction** of the speculative thread **from being forwarded to a non-speculative instruction** of a non-speculative thread **based on information associated with the speculative store instruction** of the speculative thread and **information associated with the non-speculative instruction** of the non-speculative thread," (Claim 1 emphasis added).

Chamdani's only discloses speculative stores should not irreversibly modify processor state

in order to be able to re-execute (paragraph 0031) and a description of not irreversibly modifying processor state through holding such stores in a store buffer (See paragraph 0049). And, as the Office Action states, Chamdani does not disclose blocker logic to prevent data, as in applicant's claim 1.

In addition, Hoogerbruge only discloses replacing a forwarded load instruction with a commit instruction to circumvent forwarding of subsequent store data to a forwarded load (Col. 1 lines 55-60), not preventing data based on information for the instruction as in applicant's claim 1. As can be seen from Hoogerbruge's disclosure, data for a forwarded load is replaced with store data from a previous (in program order) store instruction if there addresses are the same – the replacement of data in the register loaded by a forwarded load with store data from a previous program order store instruction. Therefore, to circumvent this same comparison and replacement between the forwarded load and a subsequent (in program order) store instruction, Hoogerbruge description of Chen includes replacing the forwarded load instruction with a commit instruction (every forwarded load instruction is replaced with an additional commit instruction col. 1 line 63). As a result, Hoogerbruge does not block store data based on information for the speculative store instruction and information for the non-speculative instruction, but rather avoids substitution in reliance on no comparison being performed between a forwarded load and a subsequent store, since there is no comparison between a commit and a store instruction. In other words, Hoogerbruge avoids any comparison between the subsequent store and forwarded load, while applicant's claim 1 specifically prevents the data based on information associated with both instructions.

Applicant's claim 11 includes, "a processor associated with the memory system, the processor including dependence blocker logic to prevent data associated with a store instruction of a speculative thread from being forwarded to an instruction of a non-speculative thread **without replacement of the instruction of the non-speculative thread**, and to allow the data associated

with the store instruction of the speculative thread to be forwarded to a speculative instruction of another speculative thread," (claim 11 emphasis added).

Similar to the discussion above, Hoogerbruge only discloses avoiding replacement from a subsequent store through addition of a commit instruction at a forwarded load instruction. However, in explicit contrast, applicant's claim 11 includes blocker logic to prevent data without such replacement – without replacement of the instruction of the non-speculative thread.

Applicant's claim 21 includes, "determining the dependence check is successful in response to determining the store address matches the load address and determining the load instruction and the in-flight store instruction each originate with a speculative thread… forwarding, if the dependence check is successful, store data associated with the in-flight store instruction to the load instruction; and declining to forward, if the dependence check is not successful, the store data to the load." As stated above, Chamdani only discloses a buffer to buffer speculative stores and a method for detecting invalidation of a speculative thread. However, Chamdani does not disclose the ability to forward speculative data or declining to forward speculative data based on whether a dependence check is successful – addresses match and each originate from a speculative thread. Specifically, Hoogerbrugge discloses avoiding address comparison of the commit instruction, which replaced the forwarded load, and a subsequent store instruction. In other words, Hoogerbrugge discloses only a first address check between a store and a load, where the store data replaces forwarded load data if the address match; however, replacement of the forwarded load with a subsequent store is not performed though a dependence check, but rather avoidance, i.e. no check at all due to the replacement of the forwarded load with a commit instruction.

Applicant's clam 26 includes, "marking the cache line as speculative." As noted in The Office Action, neither Chamdani or Hoogerbrugge disclose marking a cache line as speculative. Furthermore, Kranich only discloses marking an entry in a reorder buffer tag translation buffer

(RTB) as whether data is available or not available (Col. 12). Note two major differences: (1) Kranich does not disclose marking as speculative, but rather as data available or not available; and (2) marking is of a RTB entry, not a cache line as in applicant's claim 26.

Applicant's claim 34 includes, "control logic to prevent data associated with the store instruction from being consumed by the non-speculative thread, based on the speculation ID holding the first value." As discussed above, Hoogerbrugge only discloses replacing a forwarded load with a commit instruction to avoid dependency comparison between a subsequent store and the now added commit instruction, but never describes preventing data from being consumed by a non-speculative thread based on a speculation ID holding the first value, which indicates the store instruction is associated with the speculative thread. Also note from the discussion above that Kranich does not disclose marking as speculative or non-speculative, but rather data available or not available, and furthermore does not describe not allowing a non-speculative thread to consume based on that marking.

As a result, applicant respectfully submits that independent claims 1, 11, 21, 26, and 34, as well as their dependent claims, are now in condition for allowance for at least the reasons stated above. Furthermore, if an interview would expedite prosecution or in any way provide clarity on applicant's arguments, please feel free to contact David P. McAbee at (503) 712-4988 to schedule the interview. If there are any additional charges, please charge Deposit Account No. 50-0221.

Respectfully submitted,
Intel Corporation

Dated: May 28, 2009

/David P. McAbee/Reg. No. 58,104
David P. McAbee
Reg. No. 58,104

Intel Corporation
M/S JF3-147
2111 NE 25th Avenue
Hillsboro, OR 97124
Tele – 503-712-4988
Fax – 503-264-1729